HEX-ASCII Table

NUL	00	+	2B	V	56
SOH	01	,	2C	W	57
STX	02	–	2D	X	58
ETX	03	.	2E	Y	59
EOT	04	/	2F	Z	5A
ENQ	05	0	30	[	5B
ACK	06	1	31	\	5C
BEL	07	2	32	]	5D
BS	08	3	33	^ (↑)	5E
HT	09	4	34	_ (←)	5F
LF	0A	5	35	`	60
VT	0B	6	36	a	61
FF	0C	7	37	b	62
CR	0D	8	38	c	63
SO	0E	9	39	d	64
SI	0F	:	3A	e	65
DLE	10	;	3B	f	66
DC1 (X-ON)	11	<	3C	g	67
DC2 (TAPE)	12	=	3D	h	68
DC3 (X-OFF)	13	>	3E	i	69
DC4 (TAPE)	14	?	3F	j	6A
NAK	15	@	40	k	6B
SYN	16	A	41	l	6C
ETB	17	B	42	m	6D
CAN	18	C	43	n	6E
EM	19	D	44	o	6F
SUB	1A	E	45	p	70
ESC	1B	F	46	q	71
FS	1C	G	47	r	72
GS	1D	H	48	s	73
RS	1E	I	49	t	74
US	1F	J	4A	u	75
SP	20	K	4B	v	76
!	21	L	4C	w	77
"	22	M	4D	x	78
#	23	N	4E	y	79
$	24	O	4F	z	7A
%	25	P	50	{	7B
&	26	Q	51	|	7C
'	27	R	52	} (ALT MODE)	7D
(	28	S	53	~	7E
)	29	T	54	DEL (RUB OUT)	7F
*	2A	U	55		

intel®

3065 Bowers Avenue, Santa Clara, California 95051
(408) 987-8080

Printed in U.S.A.
A1673/286/4K/DD/KH

intel® *Software Development Tools*

MCS®-51 Macro Assembly Language Pocket Reference for DOS Systems

122755-001

TABLE OF CONTENTS

PROGRAM STATUS WORD (PSW—0D0H)

C	AC	F0	RS1	RS0	OV		P
7	6	5	4	3	2	1	0

C	Carry Flag
AC	Auxiliary Carry Flag
F0	Flag 0
RS1	Register Bank Select
RS0	Register Bank Select
OV	Overflow Flag
PSW.1	Not Used
P	Parity

OPERAND DEFINITIONS

Operand	Meaning
# data	data coded in instruction
data addr	on-chip memory address
Rr	0<=*r*<=7 General-Purpose register
@Rr	0<=*r*<=1 Indirect address register
code addr	16-bit address encoded as: full 16-bit 11-bit page address 8-bit relative offset
A	Accumulator
C	Carry flag
bit addr	bit address (on-chip)
/ bit addr	complemented contents of bit address
DPTR	Data Pointer
PC	Program Counter
AB	Register Pair

ARITHMETIC INSTRUCTIONS

Hex Code	Mnemonic	Operands
24	ADD	A,*#data*
26	ADD	A,@R0
27		R1
2*	ADD	A,R*r*
25	ADD	A,*data addr*
34	ADDC	A,*#data*
36	ADDC	A,@R0
37		R1
3*	ADDC	A,R*r*
35	ADDC	A,*data addr*
94	SUBB	A,*#data*
96	SUBB	A,@R0
97		R1
9*	SUBB	A,R*r*
95	SUBB	A,*data addr*
04	INC	A
06	INC	@R0
07		R1
0*	INC	R*r*
05	INC	*data addr*
A3	INC	DPTR
14	DEC	A
16	DEC	@R0
17		R1
1*	DEC	R*r*
15	DEC	*data addr*
A4	MUL	AB
84	DIV	AB
D4	DA	A

*See page 7 for instruction's hex code.

LOGICAL INSTRUCTIONS

Hex Code	Mnemonic	Operands
54	ANL	A,*#data*
56	ANL	A,@R0
57		R1
5*	ANL	A,R*r*
55	ANL	A,*data addr*
53	ANL	*data addr*,*#data*
52	ANL	*data addr*,A
44	ORL	A,*#data*
46	ORL	A,@R0
47		R1
4*	ORL	A,R*r*
45	ORL	A,*data addr*
43	ORL	*data addr*,*#data*
42	ORL	*data addr*,A
64	XRL	A,*#data*
66	XRL	A,@R0
67		R1
6*	XRL	A,R*r*
65	XRL	A,*data addr*
63	XRL	*data addr*,*#data*
62	XRL	*data addr*,A
E4	CLR	A
F4	CPL	A
23	RL	A
33	RLC	A
03	RR	A
13	RRC	A
C4	SWAP	A

* See page 7 for instruction's hex code.

DATA MOVE INSTRUCTIONS

Hex Code	Mnemonic	Operands
74	MOV	A,*#data*
E6	MOV	A,@R0
E7		R1
E*	MOV	A,R*r*
E5	MOV	A,*data addr*
F6	MOV	@R0,A
F7		R1
76	MOV	@R0,*#data*
77		R1
A6	MOV	@R0,*data addr*
A7		R1
F*	MOV	R*r*,A
7*	MOV	R*r*,*#data*
A*	MOV	R*r*,*data addr*
F5	MOV	*data addr*,A
75	MOV	*data addr*,*#data*
86	MOV	*data addr*,@R0
87		R1
8*	MOV	*data addr*,R*r*
85	MOV	*data addr*,*data addr*
90	MOV	DPTR,*#data* *
93	MOVC	A,@A + DPTR
83	MOVC	A,@A + PC
E0	MOVX	A,@DPTR
E2	MOVX	A,@R0
E3		R1
F0	MOVX	@DPTR,A
F2	MOVX	@R0,A
F3		R1
C0	PUSH	*data addr*
D0	POP	*data addr*
C **	XCH	A,R*r*
C5	XCH	A,*data addr*
C6	XCH	A,R0
C7		R1
D6	XCHD	A,@R0
D7		R1

* 16-bit data operand.
** See page 7 for instruction's hex code.

BIT MANIPULATION INSTRUCTIONS

Hex Code	Mnemonic	Operands
C3	CLR	C
C2	CLR	*bit addr*
D3	SETB	C
D2	SETB	*bit addr*
B3	CPL	C
B2	CPL	*bit addr*
82	ANL	C,*bit addr*
B0	ANL	C,/*bit addr*
72	ORL	C,*bit addr*
A0	ORL	C,/*bit addr*
A2	MOV	C,*bit addr*
92	MOV	*bit addr*,C

NO OPERATION

Hex Code	Mnemonic
00	NOP

PROGRAM AND MACHINE CONTROL INSTRUCTIONS

Hex Code	Mnemonic	Operands
†1	AJMP	*code addr*
B6	CJNE	@R0,#*data*,*code addr*
B7		R1
B4	CJNE	A,#*data*,*code addr*
B5	CJNE	A,*data addr*,*code addr*
B*	CJNE	Rr,#*data*,*code addr*
D*	DJNZ	Rr,*code addr*
D5	DJNZ	*data addr*,*code addr*
20	JB	*bit addr*,*code addr*
10	JBC	*bit addr*,*code addr*
40	JC	*code addr*
73	JMP	@A + DPTR
30	JNB	*bit addr*,*code addr*
50	JNC	*code addr*
70	JNZ	*code addr*
60	JZ	*code addr*
02	LJMP	*code addr*
80	SJMP	*code addr*
†1	ACALL	*code addr*
12	LCALL	*code addr*
22	RET	
32	RETI	
**	JMP	*code addr*
**	CALL	*code addr*

† First three bits of the opcode are formed by the code address operand.

* See page 7 for instruction's hex code.

** Generic form of instruction.

CODES FOR INSTRUCTIONS USING REGISTERS

	MOV Rr,A	MOV Rr, #*data*	MOV A,Rr	MOV *data addr*,Rr
R0	F8H	78H	E8H	88H
R1	F9H	79H	E9H	89H
R2	FAH	7AH	EAH	8AH
R3	FBH	7BH	EBH	8BH
R4	FCH	7CH	ECH	8CH
R5	FDH	7DH	EDH	8DH
R6	FEH	7EH	EEH	8EH
R7	FFH	7FH	EFH	8FH
	XCH A,Rr	**XRL A,Rr**	**INC Rr**	**MOV Rr, *data addr***
R0	C8H	68H	08H	A8H
R1	C9H	69H	09H	A9H
R2	CAH	6AH	0AH	AAH
R3	CBH	6BH	0BH	ABH
R4	CCH	6CH	0CH	ACH
R5	CDH	6DH	0DH	ADH
R6	CEH	6EH	0EH	AEH
R7	CFH	6FH	0FH	AFH
	ADD A,Rr	**ADDC A,Rr**	**ANL A,Rr**	**CJNE Rr, #*data*, *code addr***
R0	28H	38H	58H	B8H
R1	29H	39H	59H	B9H
R2	2AH	3AH	5AH	BAH
R3	2BH	3BH	5BH	BBH
R4	2CH	3CH	5CH	BCH
R5	2DH	3DH	5DH	BDH
R6	2EH	3EH	5EH	BEH
R7	2FH	3FH	5FH	BFH
	ORL A,Rr	**SUBB A,Rr**	**DEC Rr**	**DJNZ Rr, *code addr***
R0	48H	98H	18H	D8H
R1	49H	99H	19H	D9H
R2	4AH	9AH	1AH	DAH
R3	4BH	9BH	1BH	DBH
R4	4CH	9CH	1CH	DCH
R5	4DH	9DH	1DH	DDH
R6	4EH	9EH	1EH	DEH
R7	4FH	9FH	1FH	DFH

PREDEFINED BYTE ADDRESSES

Symbol	Hex Address	Meaning
ACC	E0	ACCUMULATOR
B	F0	MULTIPLICATION REGISTER
DPH	83	DATA POINTER *<high byte>*
DPL	82	DATA POINTER *<low byte>*
IE	A8	INTERRUPT ENABLE
IP	B8	INTERRUPT PRIORITY
P0	80	PORT 0
P1	90	PORT 1
P2	A0	PORT 2
P3	B0	PORT 3
PSW	D0	PROGRAM STATUS WORD
SBUF	99	SERIAL PORT BUFFER
SCON	98	SERIAL PORT CONTROL
SP	81	STACK POINTER
TCON	88	TIMER CONTROL
TH0	8C	TIMER 0 *<high byte>*
TH1	8D	TIMER 1 *<high byte>*
TL0	8A	TIMER 0 *<low byte>*
TL1	8B	TIMER 1 *<low byte>*
TMOD	89	TIMER MODE

PREDEFINED CODE ADDRESSES

Symbol	Hex Address	Meaning
RESET	00	Power Up (Reset)
EXTI0	03	External Interrupt 0
TIMER0	0B	Timer 0 Interrupt
EXTI1	13	External Interrupt 1
TIMER1	1B	Timer 1 Interrupt
SINT	23	Serial Port Interrupt

PREDEFINED BIT ADDRESSES

Sym.	Position	Hex	Meaning
CY	PSW.7	D7	CARRY FLAG
AC	PSW.6	D6	AUXILIARY CARRY FLAG
F0	PSW.5	D5	FLAG 0
RS1	PSW.4	D4	REGISTER BANK SELECT BIT 1
RS0	PSW.3	D3	REGISTER BANK SELECT BIT 0
OV	PSW.2	D2	OVERFLOW FLAG
P	PSW.0	D0	PARITY FLAG
TF1	TCON.7	8F	TIMER 1 OVERFLOW FLAG
TR1	TCON.6	8E	TIMER 1 RUN CONTROL BIT
TF0	TCON.5	8D	TIMER 0 OVERFLOW FLAG
TR0	TCON.4	8C	TIMER 0 RUN CONTROL BIT
IE1	TCON.3	8B	INTERRUPT 1 EDGE FLAG
IT1	TCON.2	8A	INTERRUPT 1 TYPE CONTROL BIT
IE0	TCON.1	89	INTERRUPT 0 EDGE FLAG
IT0	TCON.0	88	INTERRUPT 0 TYPE CONTROL BIT
SM0	SCON.7	9F	SERIAL MODE CONTROL BIT 0
SM1	SCON.6	9E	SERIAL MODE CONTROL BIT 1
SM2	SCON.5	9D	SERIAL MODE CONTROL BIT 2
REN	SCON.4	9C	RECEIVE ENABLE
TB8	SCON.3	9B	TRANSMIT BIT 8
RB8	SCON.2	9A	RECEIVE BIT 8
TI	SCON.1	99	TRANSMIT INTERRUPT FLAG
RI	SCON.0	98	RECEIVE INTERRUPT FLAG
EA	IE.7	AF	ENABLE ALL INTERRUPTS
ES	IE.4	AC	ENABLE SERIAL PORT INTERRUPT
ET1	IE.3	AB	ENABLE TIMER 1 INTERRUPT
EX1	IE.2	AA	ENABLE EXTERNAL INTERRUPT 1
ET0	IE.1	A9	ENABLE TIMER 0 INTERRUPT
EX0	IE.0	A8	ENABLE EXTERNAL INTERRUPT 0
PS	IP.4	BC	PRIORITY OF SERIAL PORT INTERRUPT
PT1	IP.3	BB	PRIORITY OF TIMER 1 INTERRUPT
PX1	IP.2	BA	PRIORITY OF EXTERNAL INTERRUPT 1
PT0	IP.1	B9	PRIORITY OF TIMER 0
PX0	IP.0	B8	PRIORITY OF EXTERNAL INTERRUPT 0
RD	P3.7	B7	READ DATA FOR EXTERNAL MEMORY
WR	P3.6	B6	WRITE DATA FOR EXTERNAL MEMORY
T1	P3.5	B5	TIMER/COUNTER 1 EXTERNAL FLAG
T0	P3.4	B4	TIMER/COUNTER 0 EXTERNAL FLAG
INT1	P3.3	B3	INTERRUPT 1 INPUT PIN
INT0	P3.2	B2	INTERRUPT 0 INPUT PIN
TXD	P3.1	B1	SERIAL PORT TRANSMIT PIN
RXD	P3.0	B0	SERIAL PORT RECEIVE PIN

MACRO PROCESSOR LANGUAGE (MPL) FUNCTIONS

DEFINING A FUNCTION

%*DEFINE (macro-name) (replacement-pattern)
%*DEFINE (macro-name(parameter-list)) (replacement-pattern)

MANIPULATING STRINGS

%EVAL(expression)
%LEN(string)
%SUBSTR(string,expr1,expr2)
%IN
%OUT(string)

CONTROL FUNCTIONS

%IF (expr) THEN (replacement-value) [ELSE (replacement-value)]FI
%REPEAT (expr) (replacement-value)
%WHILE (expr) (replacement-value)
%SET(name, value)
%MATCH(name1, name2) (list)
%MATCH(name1 delimiter name2) (string)

INTERPRETATION-CONTROLLING FUNCTIONS

%(balanced-text)
%nxxx...x
(n)
%EXIT
%'comment-text'
or:
%'comment-text linefeed
%METACHAR(char)

COMPARING STRINGS LEXICALLY

%EQS
%NES
%LTS
%LES
%GTS
%GES
} (string1, string2)

ASSEMBLY TIME EXPRESSION OPERATORS

In order of decreasing precedence

( )
HIGH, LOW
*, /, MOD, SHR, SHL
+, –
EQ, LT, GT, LE, GE, NE, =, <, >, <=, >=, < >
NOT
AND
OR, XOR

ASSEMBLER DIRECTIVES

Directive	Meaning
SEGMENT	Define a relocatable segment
EQU	Assign a numeric value or special assembler symbol
SET	Set symbol value
DATA	Define a data address symbol
IDATA	Define an indirect internal data address symbol
XDATA	Define an off chip data address symbol
BIT	Assign a bit address symbol
CODE	Assign a code address symbol
DS	Reserve space in byte units
DB	Insert a list of byte values
DW	Insert a list of word values
DBIT	Advance bit location counter
PUBLIC	Declare symbols outside current module
EXTRN	List symbols defined in other modules
NAME	Identify current program module
ORG	Set location counter value
END	End of program
RSEG	Select a relocatable segment
CSEG	Select an absolute segment within code address space
DSEG	Select an absolute segment within internal data address space
XSEG	Select an absolute segment within external data address space
ISEG	Select an absolute segment within indirect internal data address space
BSEG	Select an absolute segment within the bit data address space
USING	Select predefined symbolic register banks

TABLE OF INSTRUCTION OPCODES IN HEXADECIMAL ORDER

Hex Code	Number of Bytes	Mnemonic	Operands
00	1	NOP	
01	2	AJMP	*code addr*
02	3	LJMP	*code addr*
03	1	RR	A
04	1	INC	A
05	2	INC	*data addr*
06	1	INC	@R0
07	1	INC	@R1
08	1	INC	R0
09	1	INC	R1
0A	1	INC	R2
0B	1	INC	R3
0C	1	INC	R4
0D	1	INC	R5
0E	1	INC	R6
0F	1	INC	R7
10	3	JBC	*bit addr,code addr*
11	2	ACALL	*code addr*
12	3	LCALL	*code addr*
13	1	RRC	A
14	1	DEC	A
15	2	DEC	*data addr*
16	1	DEC	@R0
17	1	DEC	@R1
18	1	DEC	R0
19	1	DEC	R1
1A	1	DEC	R2
1B	1	DEC	R3
1C	1	DEC	R4
1D	1	DEC	R5
1E	1	DEC	R6
1F	1	DEC	R7
20	3	JB	*bit addr,code addr*
21	2	AJMP	*code addr*
22	1	RET	
23	1	RL	A
24	2	ADD	A,*#data*
25	2	ADD	A,*data addr*
26	1	ADD	A,@R0
27	1	ADD	A,@R1
28	1	ADD	A,R0
29	1	ADD	A,R1
2A	1	ADD	A,R2

TABLE OF INSTRUCTION OPCODES IN HEXADECIMAL ORDER (Cont'd.)

Hex Code	Number of Bytes	Mnemonic	Operands
2B	1	ADD	A,R3
2C	1	ADD	A,R4
2D	1	ADD	A,R5
2E	1	ADD	A,R6
2F	1	ADD	A,R7
30	3	JNB	*bit addr,code addr*
31	2	ACALL	*code addr*
32	1	RETI	
33	1	RLC	A
34	2	ADDC	A,*#data*
35	2	ADDC	A,*data addr*
36	1	ADDC	A,@R0
37	1	ADDC	A,@R1
38	1	ADDC	A,R0
39	1	ADDC	A,R1
3A	1	ADDC	A,R2
3B	1	ADDC	A,R3
3C	1	ADDC	A,R4
3D	1	ADDC	A,R5
3E	1	ADDC	A,R7
3F	1	ADDC	A,R7
40	2	JC	*code addr*
41	2	AJMP	*code addr*
42	2	ORL	*data addr*,A
43	3	ORL	*data addr*,*#data*
44	2	ORL	A,*#data*
45	2	ORL	A,*data addr*
46	1	ORL	A,@R0
47	1	ORL	A,@R1
48	1	ORL	A,R0
49	1	ORL	A,R1
4A	1	ORL	A,R2
4B	1	ORL	A,R3
4C	1	ORL	A,R4
4D	1	ORL	A,R5
4E	1	ORL	A,R6
4F	1	ORL	A,R7
50	2	JNC	*code addr*
51	2	ACALL	*code addr*
52	2	ANL	*data addr*,A
53	3	ANL	*data addr*,*#data*
54	2	ANL	A,*#data*
55	2	ANL	A,*data addr*

TABLE OF INSTRUCTION OPCODES IN HEXADECIMAL ORDER (Cont'd.)

Hex Code	Number of Bytes	Mnemonic	Operands
56	1	ANL	A,@R0
57	1	ANL	A,@R1
58	1	ANL	A,R0
59	1	ANL	A,R1
5A	1	ANL	A,R2
5B	1	ANL	A,R3
5C	1	ANL	A,R4
5D	1	ANL	A,R5
5E	1	ANL	A,R6
5F	1	ANL	A,R7
60	2	JZ	*code addr*
61	2	AJMP	*code addr*
62	2	XRL	*data addr*,A
63	3	XRL	*data addr*,#*data*
64	2	XRL	A,#*data*
65	2	XRL	A,*data addr*
66	1	XRL	A,@R0
67	1	XRL	A,@R1
68	1	XRL	A,R0
69	1	XRL	A,R1
6A	1	XRL	A,R2
6B	1	XRL	A,R3
6C	1	XRL	A,R4
6D	1	XRL	A,R5
6E	1	XRL	A,R6
6F	1	XRL	A,R7
70	2	JNZ	*code addr*
71	2	ACALL	*code addr*
72	2	ORL	C,*bit addr*
73	1	JMP	@A+DPTR
74	2	MOV	A,#*data*
75	3	MOV	*data addr*,#*data*
76	2	MOV	@R0,#*data*
77	2	MOV	@R1,#*data*
78	2	MOV	R0,#*data*
79	2	MOV	R1,#*data*
7A	2	MOV	R2,#*data*
7B	2	MOV	R3,#*data*
7C	2	MOV	R4,#*data*
7D	2	MOV	R5,#*data*
7E	2	MOV	R6,#*data*
7F	2	MOV	R7,#*data*
80	2	SJMP	*code addr*

TABLE OF INSTRUCTION OPCODES IN HEXADECIMAL ORDER (Cont'd.)

Hex Code	Number of Bytes	Mnemonic	Operands
81	2	AJMP	*code addr*
82	2	ANL	C,*bit addr*
83	1	MOVC	A,@A+PC
84	1	DIV	AB
85	3	MOV	*data addr*,*data addr*
86	2	MOV	*data addr*,@R0
87	2	MOV	*data addr*,@R1
88	2	MOV	*data addr*,R0
89	2	MOV	*data addr*,R1
8A	2	MOV	*data addr*,R2
8B	2	MOV	*data addr*,R3
8C	2	MOV	*data addr*,R4
8D	2	MOV	*data addr*,R5
8E	2	MOV	*data addr*,R6
8F	2	MOV	*data addr*,R7
90	3	MOV	DPTR,#*data*
91	2	ACALL	*code addr*
92	2	MOV	*bit addr*,C
93	1	MOVC	A,@A+DPTR
94	2	SUBB	A,#*data*
95	2	SUBB	A,*data addr*
96	1	SUBB	A,@R0
97	1	SUBB	A,@R1
98	1	SUBB	A,R0
99	1	SUBB	A,R1
9A	1	SUBB	A,R2
9B	1	SUBB	A,R3
9C	1	SUBB	A,R4
9D	1	SUBB	A,R5
9E	1	SUBB	A,R6
9F	1	SUBB	A,R7
A0	2	ORL	C,/*bit addr*
A1	2	AJMP	*code addr*
A2	2	MOV	C,*bit addr*
A3	1	INC	DPTR
A4	1	MUL	AB
A5		reserved	
A6	2	MOV	@R0,*data addr*
A7	2	MOV	@R1,*data addr*
A8	2	MOV	R0,*data addr*
A9	2	MOV	R1,*data addr*
AA	2	MOV	R2,*data addr*
AB	2	MOV	R3,*data addr*

TABLE OF INSTRUCTION OPCODES IN HEXADECIMAL ORDER (Cont'd.)

Hex Code	Number of Bytes	Mnemonic	Operands
AC	2	MOV	R4,*data addr*
AD	2	MOV	R5,*data addr*
AE	2	MOV	R6,*data addr*
AF	2	MOV	R7,*data addr*
B0	2	ANL	C,/*bit addr*
B1	2	ACALL	*code addr*
B2	2	CPL	*bit addr*
B3	1	CPL	C
B4	3	CJNE	A,#*data*,*code addr*
B5	3	CJNE	A,*data addr*,*code addr*
B6	3	CJNE	@R0,#*data*,*code addr*
B7	3	CJNE	@R1,#*data*,*code addr*
B8	3	CJNE	R0,#*data*,*code addr*
B9	3	CJNE	R1,#*data*,*code addr*
BA	3	CJNE	R2,#*data*,*code addr*
BB	3	CJNE	R3,#*data*,*code addr*
BC	3	CJNE	R4,#*data*,*code addr*
BD	3	CJNE	R5,#*data*,*code addr*
BE	3	CJNE	R6,#*data*,*code addr*
BF	3	CJNE	R7,#*data*,*code addr*
C0	2	PUSH	*data addr*
C1	2	AJMP	*code addr*
C2	2	CLR	*bit addr*
C3	1	CLR	C
C4	1	SWAP	A
C5	2	XCH	A,*data addr*
C6	1	XCH	A,@R0
C7	1	XCH	A,@R1
C8	1	XCH	A,R0
C9	1	XCH	A,R1
CA	1	XCH	A,R2
CB	1	XCH	A,R3
CC	1	XCH	A,R4
CD	1	XCH	A,R5
CE	1	XCH	A,R6
CF	1	XCH	A,R7
D0	2	POP	*data addr*
D1	2	ACALL	*code addr*
D2	2	SETB	*bit addr*
D3	1	SETB	C
D4	1	DA	A
D5	3	DJNZ	*data addr*,*code addr*
D6	1	XCHD	A,@R0

TABLE OF INSTRUCTION OPCODES IN HEXADECIMAL ORDER (Cont'd.)

Hex Code	Number of Bytes	Mnemonic	Operands
D7	1	XCHD	A,@R1
D8	2	DJNZ	R0,*code addr*
D9	2	DJNZ	R1,*code addr*
DA	2	DJNZ	R2,*code addr*
DB	2	DJNZ	R3,*code addr*
DC	2	DJNZ	R4,*code addr*
DD	2	DJNZ	R5,*code addr*
DE	2	DJNZ	R6,*code addr*
DF	2	DJNZ	R7,*code addr*
E0	1	MOVX	A,@DPTR
E1	2	AJMP	*code addr*
E2	1	MOVX	A,@R0
E3	1	MOVX	A,@R1
E4	1	CLR	A
E5	2	MOV	A,*data addr*
E6	1	MOV	A,@R0
E7	1	MOV	A,@R1
E8	1	MOV	A,R0
E9	1	MOV	A,R1
EA	1	MOV	A,R2
EB	1	MOV	A,R3
EC	1	MOV	A,R4
ED	1	MOV	A,R5
EE	1	MOV	A,R6
EF	1	MOV	A,R7
F0	1	MOVX	@DPTR,A
F1	2	ACALL	*code addr*
F2	1	MOVX	@R0,A
F3	1	MOVX	@R1,A
F4	1	CPL	A
F5	2	MOV	*data addr*,A
F6	1	MOV	@R0,A
F7	1	MOV	@R1,A
F8	1	MOV	R0,A
F9	1	MOV	R1,A
FA	1	MOV	R2,A
FB	1	MOV	R3,A
FC	1	MOV	R4,A
FD	1	MOV	R5,A
FE	1	MOV	R6,A
FF	1	MOV	R7,A